CLOCKING

Grimm AUDIO

www.grimmaudio.com
info@grimmaudio.com

1 Always Chasing the Clock: the PLL

Understanding how various audio devices respond to different method of clocking requires a closer look at the device that's at the heart of any synchronisation effort: the Phase Locked Loop or PLL.

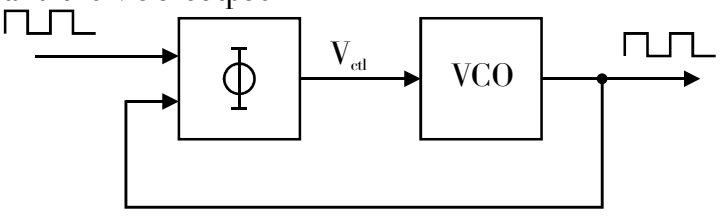
2 What is a PLL for?

Phase Locked Loops (PLL's) figure in a variety of applications in digital audio. The common thread is that a device needs to synchhronise to an external time base, but is unable to use the externally generated signal directly. A word sync signal for instance is usually at 44.1kHz or 48kHz but digital audio devices need a multiple of that, typically 256 times the sample rate. So really we want to have a clock of 11.2896MHz that runs neatly synchhronous to a 44.1kHz reference. This is called clock multiplication.

Another reason for using a PLL is when the input clock is unstable and the regenerated clock needs to be synchhronised to a time-averaged version of the input clock. An example would be a clock generated from a blackburst (video reference synchh) signal.

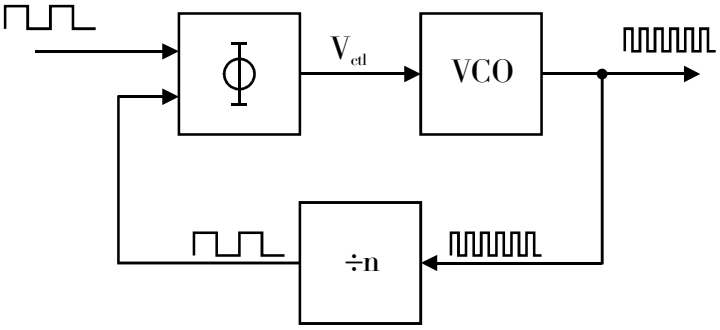
3 How do PLL's work?

The structure of a PLL is always the same. A voltage-controlled oscillator is kept in sync with the input signal by a circuit that tracks the phase difference between the input clock and the VCO output.



When the VCO runs slow, phase will start lagging compared to the input. This phase lag is converted into an increasing voltage by the phase comparator. The VCO responds to this increased control voltage by speeding up again.

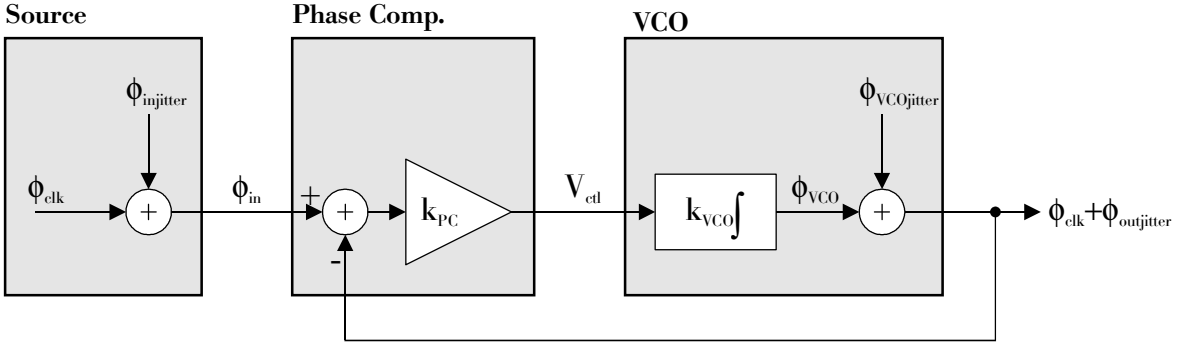
When we need a multiple of the input clock, say 256x higher, the VCO output is divided by 256 using a counter circuit. That way the phase comparator sees two clocks that have the same frequency, while the VCO actually generates a frequency that is 256 times higher.



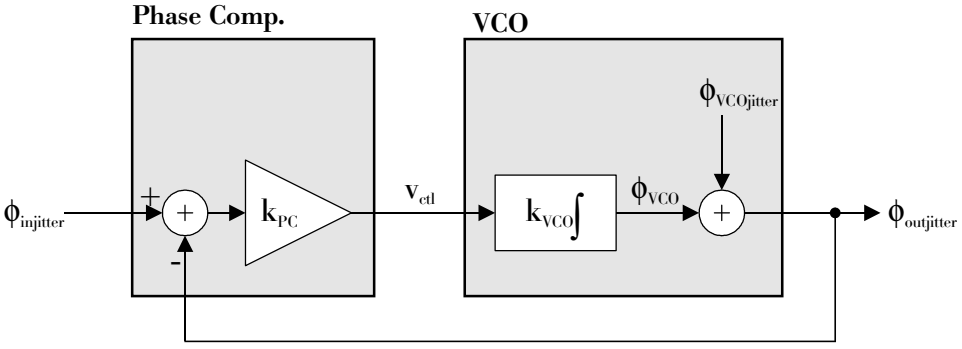
4 Some control theory

(Note: math fanatics may take issue with this exposé. Not because it's incorrect, but because I'm taking liberties with the kind of rigorous writing you'd need for a proper analysis. The reader will understand I'm choosing readability over rigour.)

To understand what a PLL does with jitter, we must do some rudimentary analysis. Below is the same block diagram as above, but the nonideal signals are represented as an ideal signal plus a jitter component.



The input signal to the PLL is a clock signal of a given frequency, with a certain amount of jitter on it. We may conceive of this jitter as a varying phase shift compared to an ideal clock. Since it's only the jitter components we're interested in, we may ignore the ϕ_{clk} terms and leave only the jitter terms in:



The phase comparator produces a control voltage proportional to the phase difference of its inputs and a constant factor k_{PC} (in volts per radian):

$$v_{ctl} = k_{PC} \cdot (\phi_{injitter} - \phi_{outjitter}) \quad (1)$$

Note that we're writing v_{ctl} with a lower case v. This is the component of V_{ctl} associated with only the jitter (remember we're ignoring the ideal part).

An oscillator produces a constantly increasing phase angle. In a VCO the rate of change of this phase angle varies with the control voltage. This means that a VCO is a phase integrator:

$$\phi_{VCO} = k_{VCO} \cdot \int v_{ctl} \cdot dt \tag{2}$$

where k_{VCO} is the product of the tuning factor of the oscillator (specified in %/V or ppm/V) and the centre frequency of the VCO. Before we land ourselves into differential equation domain, let's Laplace transform equation 2:

$$\phi_{VCO} = v_{ctl} \cdot \frac{k_{VCO}}{s} \tag{3}$$

Wrapping the whole loop together yields:

$$\phi_{outjitter} = \phi_{VCOjitter} + \left(\phi_{injitter} - \phi_{outjitter}\right) \cdot \frac{k_{PC} \cdot k_{VCO}}{s} \tag{4}$$

Solving for output jitter we get:

$$\phi_{outjitter} = \frac{s}{s + k_{PC} \cdot k_{VCO}} \cdot \phi_{VCOjitter} + \frac{k_{PC} \cdot k_{VCO}}{s + k_{PC} \cdot k_{VCO}} \cdot \phi_{injitter} \tag{5}$$

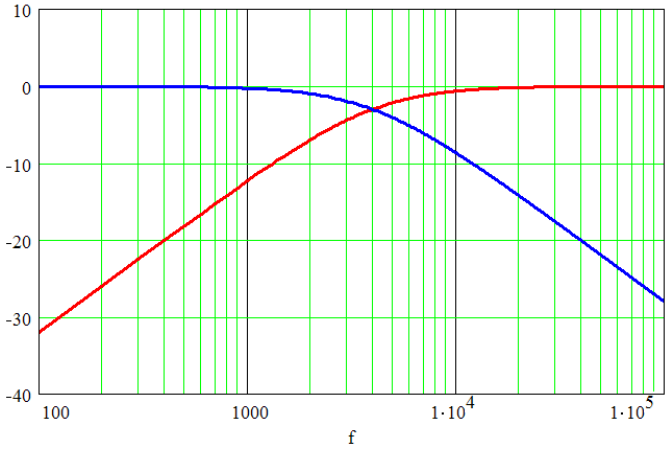
Phase at the output of the PLL is a mix of the input jitter and the VCO jitter. Crucially, both are filtered differently. The factor of $\phi_{VCOjitter}$ is a high pass filter. The factor of $\phi_{injitter}$ is a low pass filter. Both have the same pole: $k_{PC} \cdot k_{VCO}$. That is to say, both have a corner frequency of $k_{PC} \cdot k_{VCO}/(2 \cdot \pi)$.

5 The PLL as a crossover filter

Analysis of this simplest PLL already tells us the most important story: at low frequencies, jitter from the input signal is passed unattenuated, while jitter from the VCO is strongly reduced. At high frequencies, jitter from the VCO is passed unattenuated while jitter from the input is reduced.

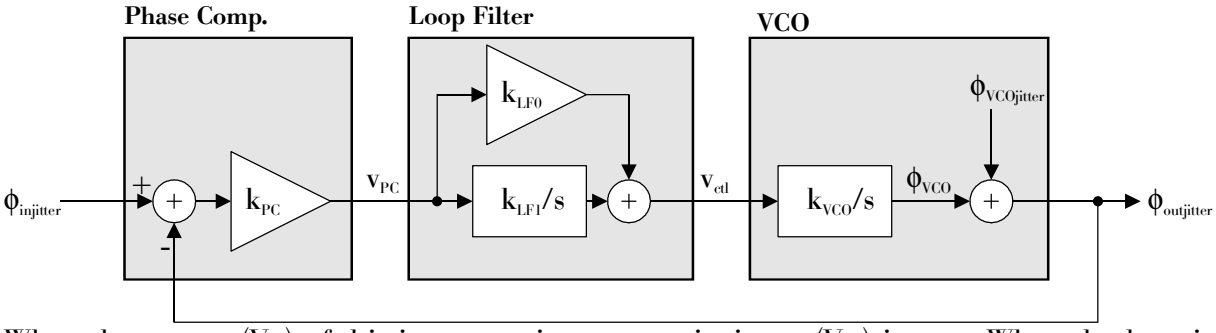
The confusing bit is that in this context, "frequency" is not the actual oscillator frequency but "jitter frequency" that is, how quickly the phase wobbles back and forth compared to an ideal clock. The preceding paragraph says that if the input clock slowly rocks from fast to slow, the VCO will be made to follow this. If the input rapidly shudders, the VCO will only follow the average frequency, not the quick excursions. On the other hand, if the VCO itself has a tendency to drift slowly, the PLL will compensate for this, forcing the VCO to keep a steady lock with the input. If the VCO rapidly jitters, there will be little counteraction. Somewhere between fast and slow there is a crossover point.

Here's the transfer of a basic PLL, designed to have a corner frequency of 4kHz. The blue curve shows the attenuation of input jitter. The red curve shows the attenuation of VCO jitter.



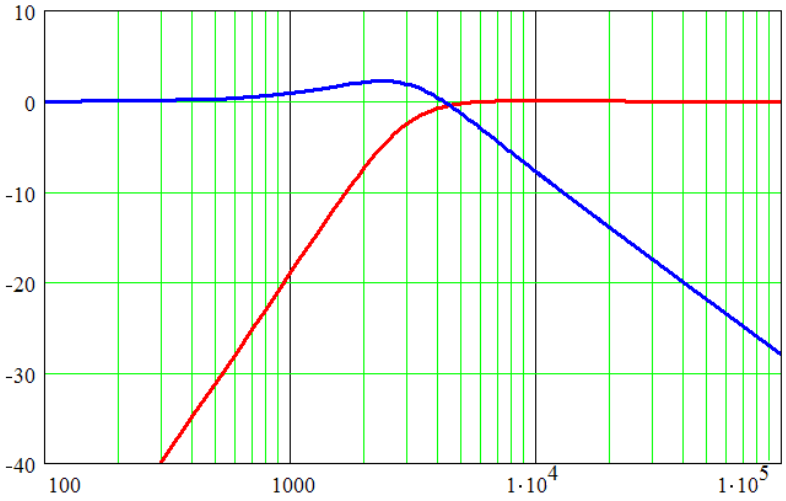
The PLL of the preceding analysis has some shortcomings. Math-heads will already have noticed that if the integrator was to produce a constantly increasing phase, there had to be a constant at the input. What this means in practice is this: a VCO produces its "center frequency" when the control input is zero volts. Higher or lower frequencies are produced for positive or negative control voltages respectively. That means that when we're locking the PLL to a frequency other than the exact center frequency of the VCO, there has to be a nonzero control voltage going into the VCO. Now the phase comparator produces a voltage that's proportional to the phase error. So, to keep the PLL locked to a frequency that isn't exactly the VCO's center frequency, there has to be a definite phase error at the input of the phase comparator! Meaning a PLL as simple as this will produce a phase-shifted output for all but one frequency.

Practical PLL's will therefore have at least one extra integrator in the loop:



When the output (V_{ctl}) of this integrator is constant, its input (V_{PC}) is zero. When the loop is in lock, there is no phase difference between input and output.

The transfer function of input jitter to output and of VCO jitter to output will be different now:



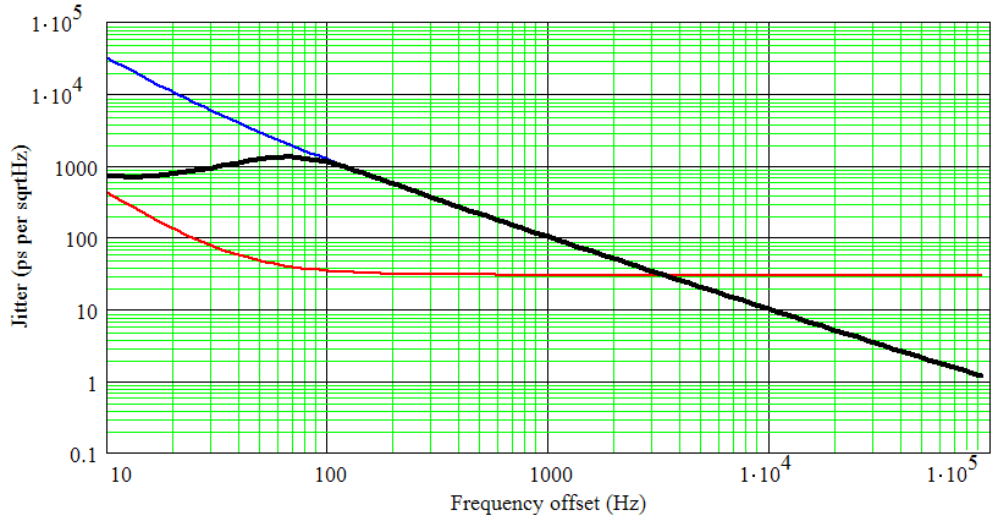
This is the transfer found in the simplest (and most common) practical PLL's. The VCO's own jitter is strongly attenuated (40dB/dec) towards low frequencies, the incoming jitter is slightly boosted around the corner frequency and is then attenuated at a rate of 20dB/dec.

The jitter boost around the corner frequency is, alas, unavoidable. This why AES3 daisy chains without house sync are so prone to jitter accumulation. It's not the addition of jitter that causes trouble, it's the gain.

6 Design practice

Choosing the corner frequency given the VCO design

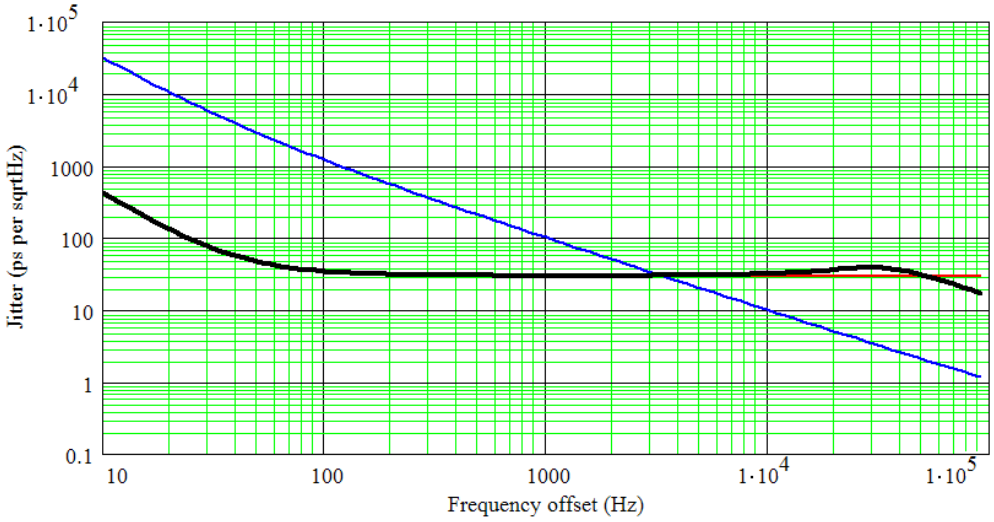
The right choice of corner frequency is dictated by the amount of jitter one expects to find at the input and by the jitter of the VCO itself. Sometimes a low PLL corner frequency is used as a sales argument. Without knowledge of the performance of the VCO this is quite meaningless. A typical example would be a simple AES/EBU receiver. The VCO on the receiver has quite acceptable short-term jitter but a lot of low-frequency jitter. This is locked to an AES/EBU signal with good long-term stability but with a lot of wideband noise on it. If the PLL corner were set to a low frequency, the result would be disastrous:



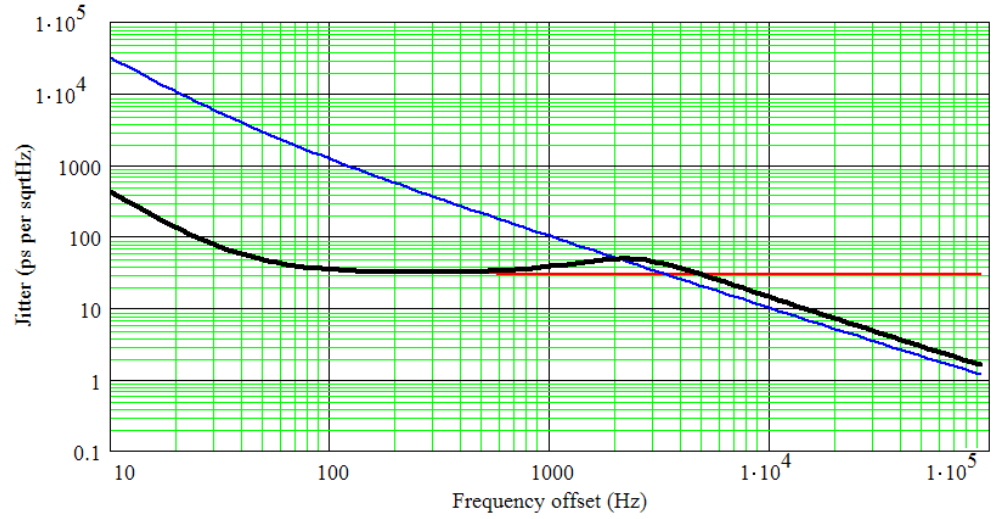
Red shows the jitter on the AES/EBU signal (I should point out that this graph oversimplifies matters a bit, representing the wideband jitter on an AES/EBU as random white noise.

In reality that's not quite the case of course). Blue shows the jitter spectrum of the VCO. Black is the output of the PLL when the loop filter is set for a crossover frequency of 100Hz. Clearly this is not optimal: below 3.3kHz the PLL actually adds jitter, and a fair amount of it. A low PLL corner frequency alone is not necessarily a good thing.

Too high a corner frequency isn't good either:



The VCO is made to follow all of the jitter of the input signal up to about 50kHz, even though left alone it would do better starting from some 3.3kHz. You see the conclusion coming: the optimal corner frequency for a given PLL and a given input signal is where the two spectra cross:



Overall, the resulting signal has less jitter than either the input signal or the VCO alone.

Unfortunately the designer can't know how noisy an input signal to expect. Nevertheless, just having a good estimate and following that will result in a much better PLL design than one with an arbitrarily chosen corner frequency.

Choosing the VCO design given the corner

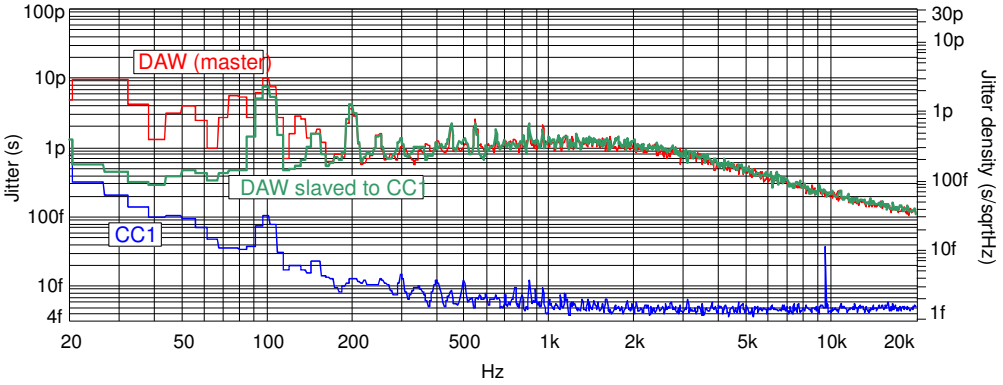
The argument can be turned around. Sometimes conditions impose a particular corner frequency. More specifically, if a PLL is to be used for cleaning clocks, the corner frequency is no longer free to choose. In that case, optimising the design means designing a VCO with

the noise performance to go with it. This is the case with the CC1. The corner frequency has been set very low, around 0.1Hz. In order to be certain that the regenerated clock is always better than what came in, the noise specification for the VCO becomes extremely stringent.

A PLL intended for clock recovery will have a fast PLL, so a basic VCO will do. A PLL intended for jitter removal will have a slow PLL, and the VCO will need to have markedly lower jitter than what is expected at the input.

7 Practical example

A neat example of the theory is a measurement made on the clock input of the ADC chips in a popular DAW system.



The red curve shows the jitter spectrum in master mode. Note that we already see the two-corner spectrum of a PLL system. This indicates that even in master mode, there's a PLL in the circuit. The system simply switches between an internal crystal-derived word sync reference and an external one.

The PLL corner frequency is around 4kHz. As the transfer graphs predicted, the PLL attenuates VCO jitter at a rate of 40dB/dec. Now, the VCO jitter density slopes upward toward low frequencies at a rate of just less than 40dB/dec, so as the PLL fights down, the VCO fights up. Nevertheless, the VCO is substantially tamed, to such an extent that below 200Hz the crystal oscillator actually catches up with it. This is quite an interesting effect. It means that if an external oscillator is better than the one built in by the manufacturer, low-frequency jitter actually improves in slave mode. This is demonstrated by locking the device to the CC1. The jitter spectum of the CC1 is shown in blue. The result is shown in green.

The commonly held intuitive notion that internal clocking must be better than external clocking is simply not always true. It is quite common for at least low frequency jitter performance to improve markedly with external clocking. The only reason why this is so little known is because the jitter test capability of common audio test kit is insufficient to resolve the noise from even mediocre crystal oscillators.

Under these conditions it would have been better had the PLL corner frequency been much higher, but this product was designed to work well even when locked to word sync signals much, much noisier than the CC1. Still, a user-settable PLL would've been nice here.

8 Conclusions

1. If, and only if, the VCO is extremely stable, it makes sense to place the corner as low as possible. When the VCO is noisy, it's better to place the corner frequency higher up. That way even a noisy VCO can take advantage of a quiet external reference.
2. Conversely, slow PLL's require oscillators with excellent long-term stability while fast PLL's will do nicely with simple VCOs.
3. Locking a device to an external clock that is more stable than its internal oscillator always improves low-frequency jitter.
4. In equipment where the PLL is always on, external clocking yields an net improvement. Jitter will be lower at low frequencies while high-frequency jitter performance will not deteriorate.